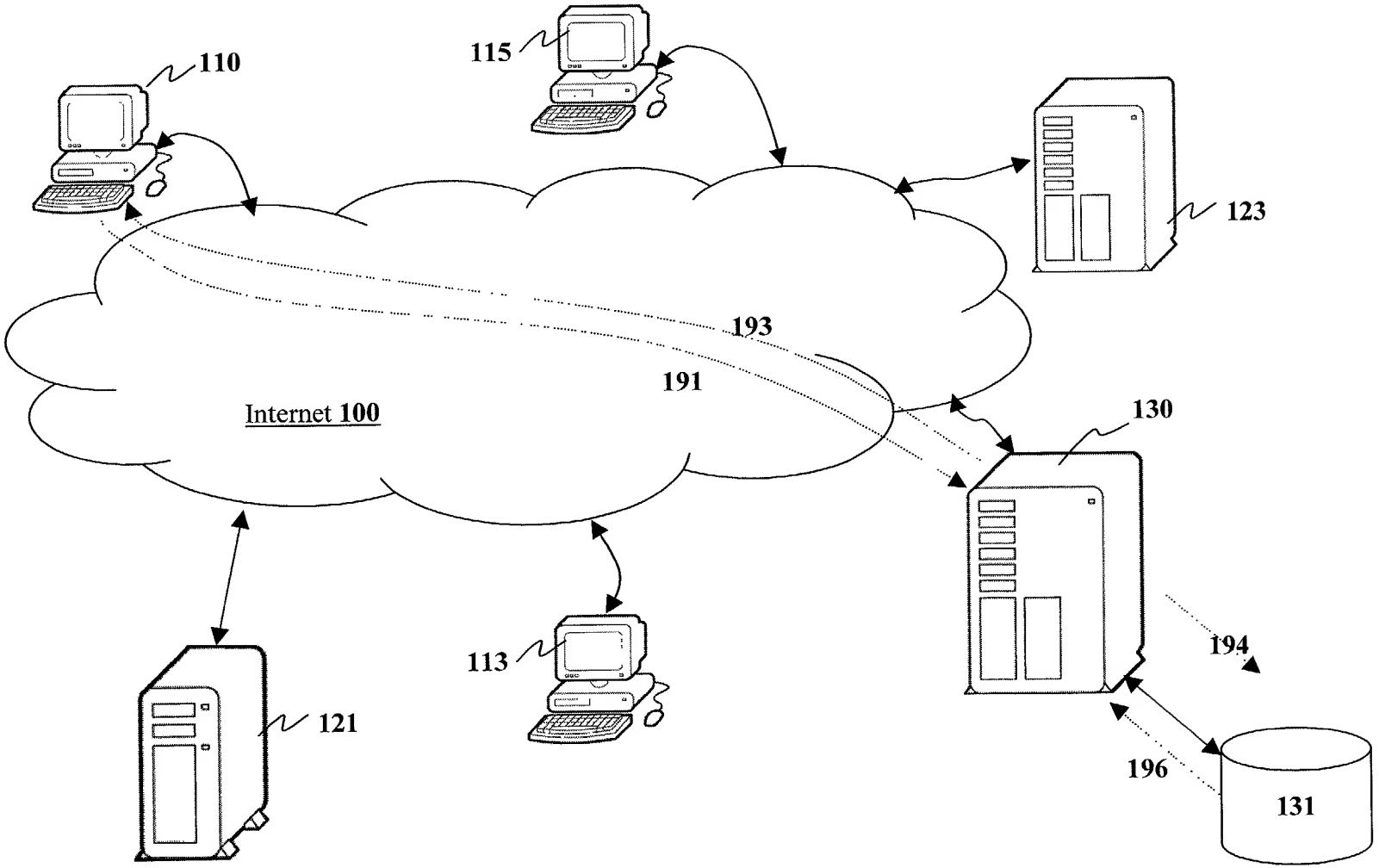
Page 2 of 3

Figure 1



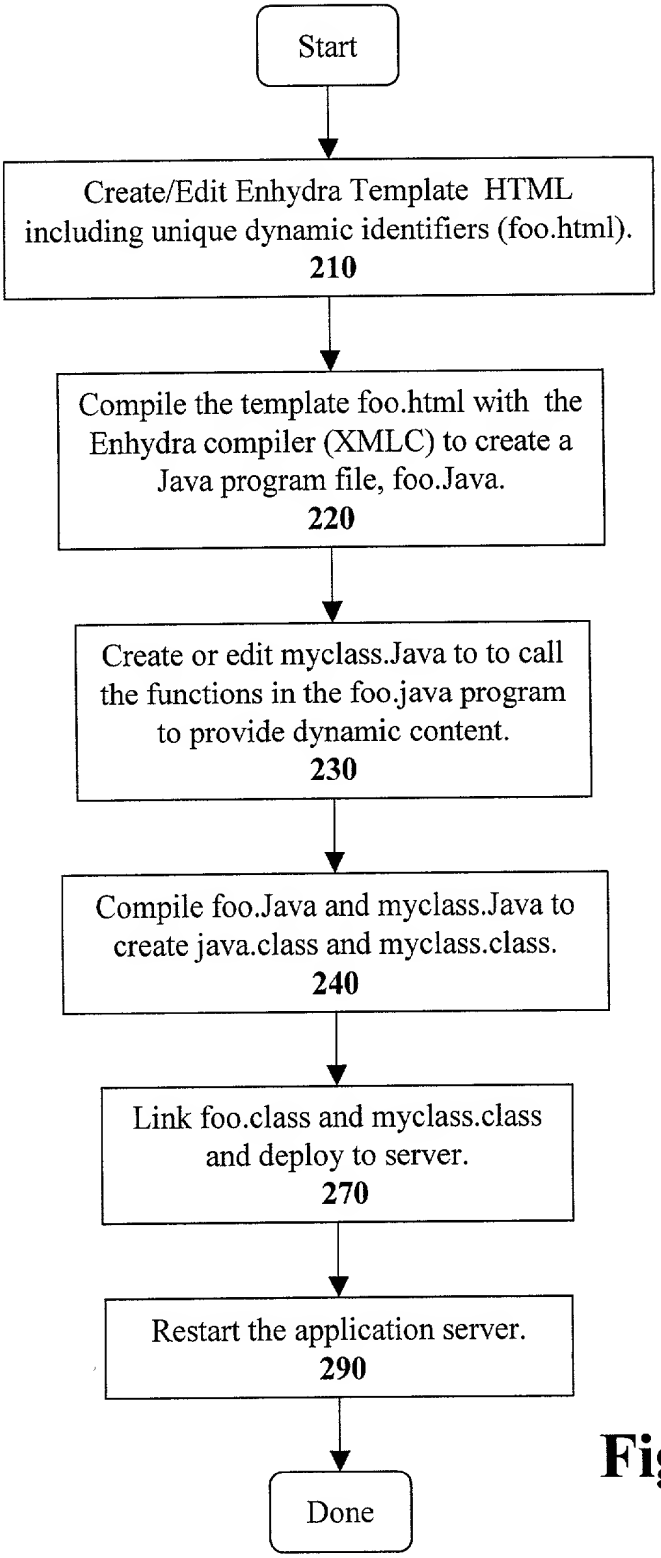

Start
Create/Edit Enhydra Template HTML including unique dynamic identifiers (foo.html).
210
Compile the template foo.html with the Enhydra compiler (XMLC) to create a Java program file, foo.Java.
220
Create or edit myclass.Java to to call the functions in the foo.java program to provide dynamic content.
230
Compile foo.Java and myclass.Java to create java.class and myclass.class.
240
Link foo.class and myclass.class and deploy to server.
270
Restart the application server.
290
Done

Figure 2

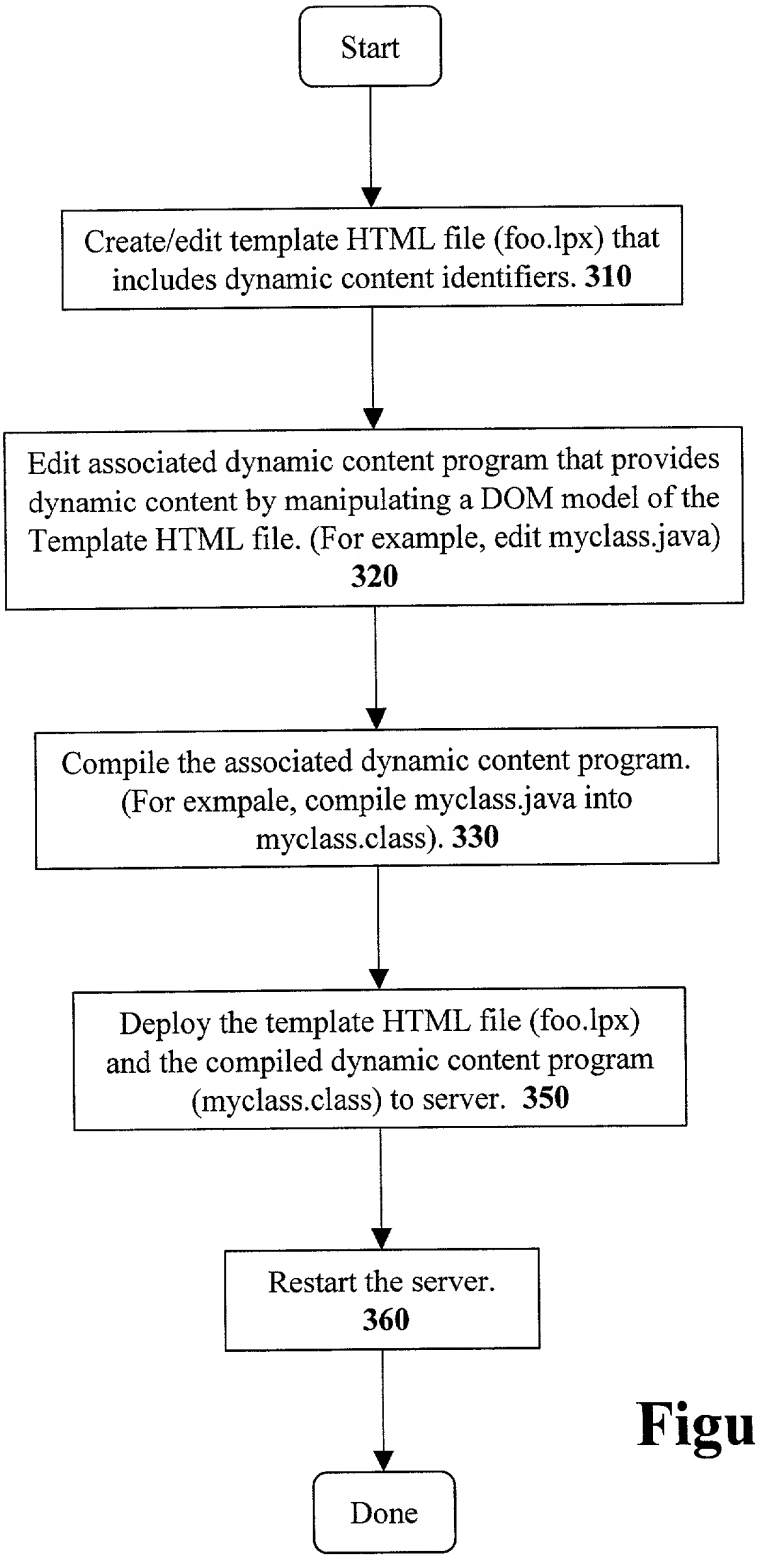

Start
Create/edit template HTML file (foo.lpx) that includes dynamic content identifiers. 310
Edit associated dynamic content program that provides dynamic content by manipulating a DOM model of the Template HTML file. (For example, edit myclass.java) 320
Compile the associated dynamic content program. (For exmpale, compile myclass.java into myclass.class). 330
Deploy the template HTML file (foo.lpx) and the compiled dynamic content program (myclass.class) to server. 350
Restart the server. 360
Done

Figure 3

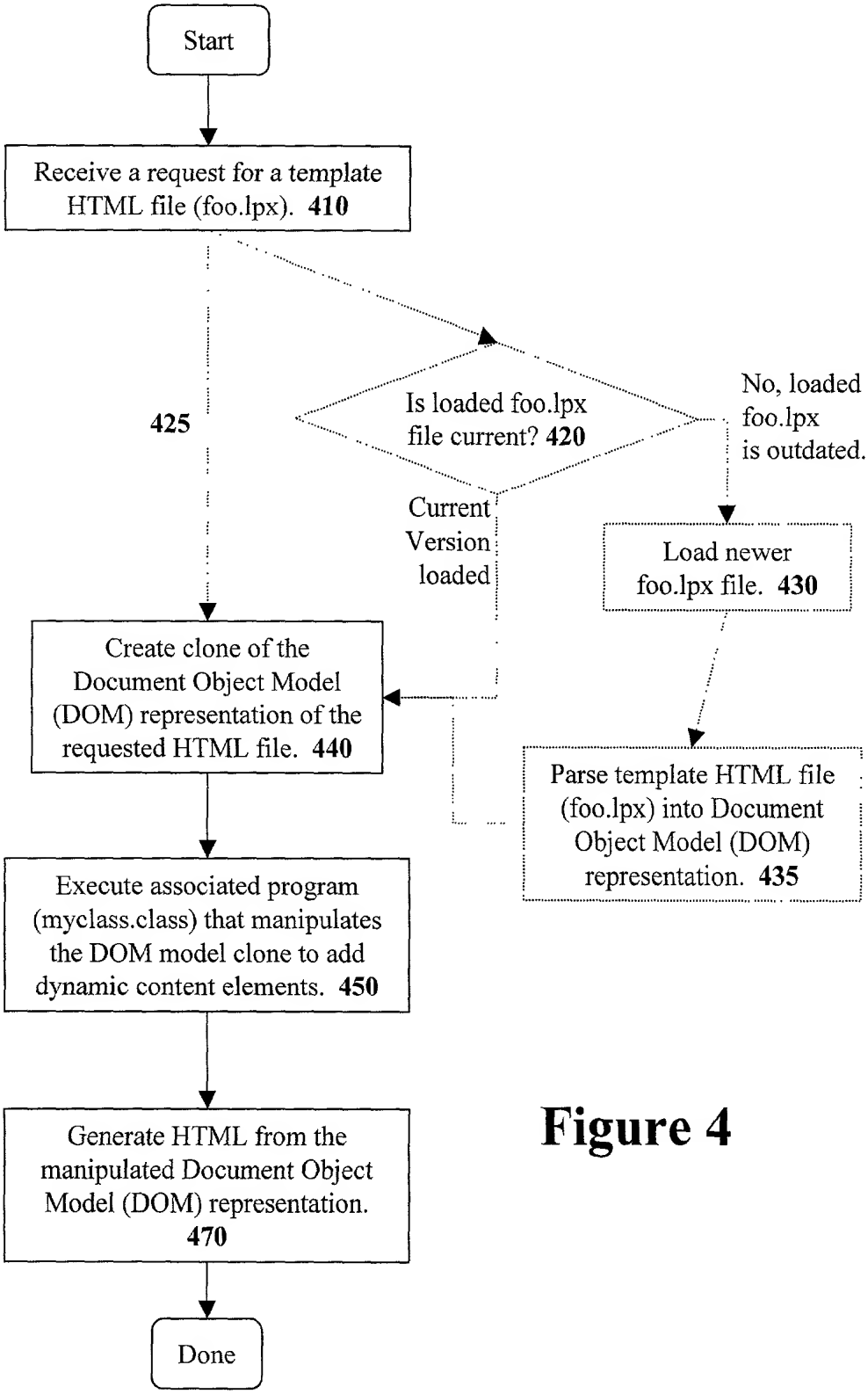

Start
Receive a request for a template HTML file (foo.lpx). 410
425
Is loaded foo.lpx file current? 420
No, loaded foo.lpx is outdated.
Current Version loaded
Load newer foo.lpx file. 430
Create clone of the Document Object Model (DOM) representation of the requested HTML file. 440
Parse template HTML file (foo.lpx) into Document Object Model (DOM) representation. 435
Execute associated program (myclass.class) that manipulates the DOM model clone to add dynamic content elements. 450
Generate HTML from the manipulated Document Object Model (DOM) representation. 470
Done

Figure 4

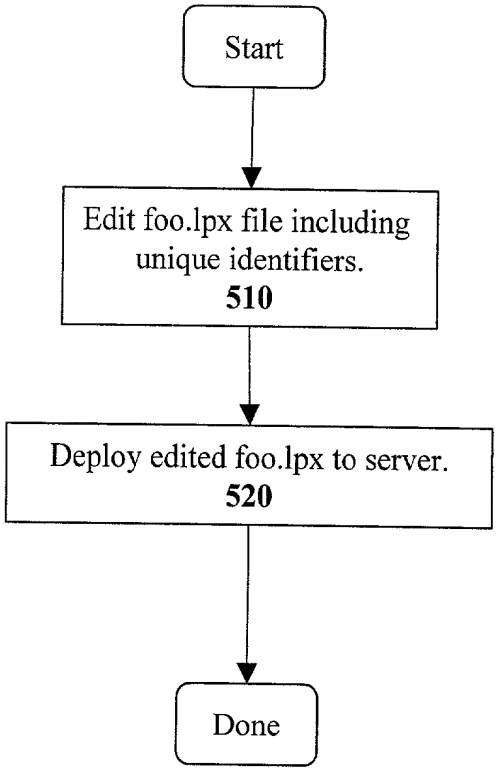

Start
Edit foo.lpx file including unique identifiers.
510
Deploy edited foo.lpx to server.
520
Done

Figure 5

Type of web site modification.	Steps to effectuate web site modification with Enhydra.	Steps to effectuate web site modification with present invention.
Simple change to static template HTML file.	1) Change the template HTML file. 2) Recompile the changed template HTML file with XMLC to create foo.java 3) Compile foo.java into the foo.class; 4) Link the changed foo.class with myclass.class 5) Deploy linked file to the server. 6) Restart the server.	1) Change the template HTML file. 2) Deploy changed template HTML file to the server.
Deletion of existing dynamic element.	1) Change the template HTML file to delete element. 2) Recompile the changed template HTML file with XMLC to create foo.java 3) Compile foo.java into the foo.class; 4) Revise the myclass.java to remove deleted dynamic content portions 5) Compile the myclass.java file into myclass.class 6) Link the changed foo.class with the myclass.class 7) Deploy linked file to the server. 8) Restart the server.	1) Change the template HTML file to delete dynamic element. 2) Deploy changed template HTML file to the server.
Addition of a new dynamic element.	1) Change the template HTML file to add element. 2) Recompile the changed template HTML file with XMLC to create foo.java 3) Compile foo.java into the foo.class; 4) Revise the myclass.java to add new dynamic content 5) Compile the myclass.java file into the myclass.class; 6) Link the changed foo.class with the myclass.class 7) Deploy linked file to the server. 8) Restart the server.	1) Change the template HTML file to add element. 2) Revise the myclass.java to add new dynamic content changes. 3) Compile the myclass.java file into the myclass.class; 4) Deploy changed foo.class and changed template HTML file to the server. 5) Restart the server.

Figure 6